

Avant-propos

1. Introduction	13
2. Parti pris de ce livre.	15
2.1 Ce qui ne sera pas abordé.	15
2.2 Choix techniques	15
2.3 Choix d'écriture	16

Chapitre 1 Introduction

1. Des applications accessibles.	17
1.1 Qu'est-ce que l'accessibilité numérique ?	18
1.2 Quatre formes de handicap	18
1.2.1 Le handicap auditif.	19
1.2.2 Le handicap moteur	20
1.2.3 Le handicap cognitif	21
1.2.4 Le handicap visuel.	23
1.3 Responsabilités des développeurs et développeuses.	24
1.3.1 Les référentiels d'accessibilité.	25
1.3.2 Savoir utiliser Talkback	27
2. Des applications modernes	32
2.1 Des challenges uniques spécifiques au mobile	32
2.2 Des appareils de plus en plus puissants menant à des interfaces de plus en plus complexes	33
2.3 Un UI Toolkit original qui n'est plus à la hauteur.	35
3. Prérequis	35
3.1 Kotlin.	35
3.1.1 Fonctions.	36
3.1.2 Traitement asynchrone	38
3.2 Android	42

2 _____ Jetpack Compose

Développez des interfaces modernes et accessibles pour Android

Chapitre 2

La genèse de Jetpack Compose

1. Introduction	45
2. Les raisons motivant la création de Jetpack Compose	45
2.1 Pouvoir mettre à jour le UI toolkit en dehors du cycle de mise à jour de la plateforme	45
2.2 Améliorer la maintenabilité du UI Toolkit original	46
2.3 Se débarrasser des choix d'implémentation contraignants, liés au principe d'héritage.	47
2.4 Simplifier le développement de composants réutilisables	48
2.5 Prendre en compte l'accessibilité dès la conception	49
2.6 Appliquer une meilleure séparation des responsabilités	52
2.7 Les lacunes de la programmation impérative	54
2.8 Conclusion	56
3. Les partis pris techniques de Jetpack Compose	57
3.1 Avant-propos	57
3.2 La structure d'un composant Jetpack Compose	58
3.3 Un composant est une fonction	58
3.3.1 La piste écartée de l'utilisation des classes	59
3.3.2 L'utilisation de fonction.	61
3.4 Un composant est une fonction qui prend en entrée des paramètres.	62
3.4.1 Des données qui circulent dans un sens	62
3.4.2 Des événements qui circulent dans l'autre sens	63
3.4.3 Une fonction Composable ne gère pas l'état du composant mais le représente	64
3.5 L'annotation @Composable	66
3.5.1 Comment fonctionne cette annotation ?	66
3.5.2 Comment utiliser le mot-clé @Composable ?	67
3.5.3 L'arbre de composition.	68

3.6	La recomposition	68
3.6.1	Les optimisations du processus de recomposition	70
3.6.2	Règle n°1 : une fonction Composable ne doit pas générer d'effet de bord	70
3.6.3	Règle n°2 : une fonction Composable doit être idempotente	73
3.7	Le nommage des fonctions Composables	75
3.8	Une fonction @Composable ne retourne rien	76
3.9	La programmation déclarative	76
4.	L'arbre sémantique pour gérer l'accessibilité	78
4.1	Représentation de l'arbre sémantique	79
4.2	Propriétés sémantiques	81
4.3	Propriétés sémantiques des composants de la bibliothèque Compose UI	83
4.4	Visualiser l'arbre sémantique et ses propriétés avec Android Studio	84
5.	De l'annotation @Composable au rendu visuel à l'écran	86
5.1	Interprétation de l'annotation Composable par le compilateur	87
5.2	Mise en mémoire de l'interface dans la Slot table lors de l'exécution	88
5.3	Les trois phases de rendu d'un écran	90
6.	Conclusion	91

Chapitre 3

Les bases de Compose UI

1.	Premiers pas	93
1.1	Les dépendances Jetpack Compose	93
1.2	Créer un nouveau projet Android avec Jetpack Compose	94
1.3	Ajouter Jetpack Compose à un projet existant	96
1.4	Le composant racine	98
1.4.1	Une activité Jetpack Compose	98
1.4.2	Jetpack Compose au sein d'une vue XML	100

4 Jetpack Compose

Développez des interfaces modernes et accessibles pour Android

1.5	Aperçu d'un composant utilisant Compose UI	102
1.6	Prévisualiser un composant avec Android Studio	104
2.	Modifier.	106
2.1	Définition	106
2.1.1	À quoi sert un Modifier ?	107
2.1.2	Les Modifiers peuvent se chaîner	108
2.1.3	L'ordre des Modifiers a une importance	109
2.1.4	Certains Modifiers sont type safe	111
2.2	Le Modifier <code>.semantics()</code> pour gérer l'accessibilité	113
2.3	Des Modifiers qui s'enchaînent	114
2.4	Bonne pratique : tout composant doit exposer un paramètre Modifier	117
3.	Les thèmes.	121
3.1	Création d'un <code>MaterialTheme</code>	121
3.1.1	La notion de thème au sein de Material Design	121
3.1.2	La fonction <code>ComposableMaterialTheme</code>	124
3.1.3	Définir un thème propre à notre application	125
3.1.4	Personnalisation des couleurs	127
3.1.5	Personnalisation de la typographie	128
3.1.6	Personnalisation des formes.	133
3.2	L'objet <code>MaterialTheme</code> pour accéder aux attributs de notre thème.	134
3.3	Appliquer les couleurs "On" par défaut.	136
3.3.1	Le composant <code>Surface</code>	137
3.3.2	<code>CompositionLocal</code>	139
3.4	Bonnes pratiques d'accessibilité.	141
3.4.1	Des couleurs avec des niveaux de contraste suffisamment prononcés	142
3.4.2	Une typographie lisible	145
3.4.3	Un thème homogène et cohérent	146
3.4.4	Un thème sombre.	146
3.5	Aller plus loin avec les thèmes.	147
4.	Conclusion	148

Chapitre 4 Textes, boutons et images

1. Textes	149
1.1 Les paramètres de la fonction Composable Text	150
1.2 Style.....	153
1.2.1 Première version brute du composant.....	153
1.2.2 Version stylisée du composant	155
1.2.3 Version stylisée via TextStyle	156
1.2.4 Version reposant sur le thème Material Design	157
1.3 Plusieurs styles au sein d'un même texte	158
1.4 Bonnes pratiques d'accessibilité concernant les textes.....	161
1.4.1 Le contexte du texte est véhiculé implicitement par un élément non textuel	162
1.4.2 Le texte contient du texte barré	166
1.4.3 Le texte contient des caractères spéciaux ou des abréviations.....	168
1.4.4 Le texte est réparti sur plusieurs composants agencés d'une manière spécifique	168
2. Boutons	171
2.1 Les paramètres du composant Button.....	172
2.1.1 Introduction	172
2.1.2 Un composant 100 % personnalisable	175
2.1.3 Personnaliser l'apparence d'un bouton avec ButtonDefaults.....	178
2.2 Bouton icône.....	181
2.2.1 Les paramètres du composant IconButton.....	181
2.2.2 Le composant Icon	183
2.3 Boutons flottants	183
2.4 Un composant cliquable grâce au Modifier .clickable	186
2.5 Bonnes pratiques d'accessibilité concernant les boutons.....	188
2.5.1 Associer un texte alternatif aux composants de type IconButton.....	189

6 Jetpack Compose

Développez des interfaces modernes et accessibles pour Android

2.5.2	Veillez à ce que les boutons flottants soient bien atteignables avec un lecteur d'écran.	190
2.5.3	Annoncer les informations affichées temporairement suite à une action	194
2.5.4	Annoncer les changements d'état d'un composant suite à une action	198
2.5.5	Personnaliser le nom de l'action associée au bouton avec la propriété <code>onClickLabel</code>	200
3.	Images	202
3.1	Les paramètres de la fonction <code>Image</code>	202
3.2	Charger une image à partir d'une URL	205
3.3	La fonction <code>Icon</code>	206
3.4	Bonnes pratiques d'accessibilité.	208
3.4.1	Quand faut-il écrire un texte alternatif ?	208
3.4.2	Comment écrire un texte alternatif ?	210

Chapitre 5

Champs de formulaire

1.	Introduction	213
2.	Champs de saisie.	213
2.1	Aperçu du composant <code>TextField</code>	214
2.2	Les paramètres de la fonction <code>Composable TextField</code>	217
2.3	Style.	220
2.4	Styles multiples au sein d'un champ de saisie	222
2.5	Faciliter la saisie des douze chiffres d'une carte bancaire.	225
2.6	Bonnes pratiques d'accessibilité pour les champs de saisie	227
2.6.1	Donner la possibilité à l'utilisateur d'afficher et masquer le contenu d'un champ de type mot de passe	227
2.6.2	Informé suffisamment tôt l'utilisateur des contraintes d'un champ de saisie.	232
2.6.3	Informé l'utilisateur s'il y a une erreur dans sa saisie	234
2.6.4	Permettre le remplissage automatique dans certains types de champs	237

2.6.5 Utiliser un label et pas seulement un placeholder	240
3. Boutons de type radio, cases à cocher et switches	245
3.1 Boutons de type radio	245
3.2 Cases à cocher	249
3.3 Switch	253
3.3.1 Amélioration du contraste du composant Switch	254
3.3.2 Associer un texte à un Switch	256

Chapitre 6

Mise en page standard

1. Introduction	259
2. Le processus de mise en page	260
2.1 La phase de layout	261
2.1.1 Le processus de mesure	261
2.1.2 Le processus de positionnement	264
3. Mise en page en ligne ou en colonne	265
3.1 Le layout Row	265
3.2 Le layout Column	267
4. Modifiers affectant la mise en page	269
4.1 La taille	270
4.2 Le positionnement	272
4.2.1 Alignement	272
4.2.2 Alignement sur la ligne de base	274
4.2.3 Poids	277
4.2.4 Marge	279
4.3 Le défilement	280
5. Mise en page en superposition	282
5.1 Les paramètres du layout Box	282
5.2 Cas d'usage du layout Box	286
5.3 Alignement unitaire des composants en superposition	288
5.4 Taille relative d'un enfant	289

8 Jetpack Compose

Développez des interfaces modernes et accessibles pour Android

6.	Mise en page selon l'espace disponible avec BoxWithConstraints	292
6.1	Cas d'usage	293
6.2	Impact sur le processus de mise en page	295
7.	Les mesures intrinsèques	296
7.1	Introduction	296
7.2	Définition des mesures intrinsèques	297
7.3	Cas d'usage en largeur	298
7.4	Cas d'usage en hauteur	300
8.	Bonnes pratiques d'accessibilité	302
8.1	Redimensionnement sans dégradation jusqu'à 200 %	302
8.1.1	Exemple	303
8.1.2	Comment détecter ce type d'anomalies ?	305
8.1.3	Correction	307
8.2	Navigation facilitée grâce aux titres	309
8.2.1	Exemple	309
8.2.2	Correction	310
8.3	Navigation facilitée grâce aux regroupements de composant	311
8.3.1	Exemple	311
8.3.2	Correction	311

Chapitre 7

Gestion des états et des effets

1.	Qu'est-ce qu'un état ?	313
1.1	Introduction	313
1.2	Différents types d'états	315
1.3	Production d'un Screen UI State	318
1.4	Production d'un UI Element State	321
1.4.1	Cas n°1 : la logique UI est simple	323
1.4.2	Cas n°2 : la logique UI est complexe et implique un ou plusieurs états	324
1.5	Résumé	329

2. Composant sans état interne	330
2.1 Origine de cette pratique	330
2.2 Conception de composants sans état	332
2.2.1 Unidirectional Data Flow (UDF).	332
2.2.2 États et événements dans les paramètres	333
2.3 Résumé	341
3. Composants avec état interne	342
3.1 La fonction remember	342
3.2 Composant avec état immutable	343
3.2.1 Cas pratique	343
3.3 Composant avec état mutable.	345
3.3.1 La fonction mutableStateOf()	345
3.3.2 Cas pratique	346
3.3.3 Différentes manières de déclarer un état mutable	347
3.4 La fonction rememberSaveable	349
3.5 Autres états Android supportés.	349
3.5.1 Cas pratique	350
3.6 Résumé	351
4. Arbre de décision : où gérer l'état ?	351
5. Gestion des effets de bord	352
5.1 Introduction	352
5.2 Gestion d'effet de bord synchrone.	354
5.2.1 DisposableEffect	354
5.2.2 SideEffect	356
5.3 Gestion d'effet de bord asynchrone	359
5.3.1 rememberCoroutineScope	359
5.3.2 LaunchedEffect	361
5.4 Effets de bord produisant un état	363
5.4.1 rememberUpdatedState	363
5.4.2 produceState	365
5.4.3 derivedStateOf	366
5.4.4 remember	368

Chapitre 8

Les listes

1. Introduction	371
2. Définition du contenu via un DSL	374
2.1 DSL : définition	374
2.2 Accès au DSL via LazyListScope	375
2.3 Les fonctions du DSL	377
2.3.1 La fonction item	377
2.3.2 Les fonctions items	379
2.3.3 La fonction itemsIndexed	384
2.3.4 La fonction stickyHeader	386
3. LazyColumn et LazyRow	387
3.1 Introduction	387
3.2 Paramètres	388
3.3 Cas d'usage	390
3.4 Point de vigilance : imbrication de Lazy layout dans un layout standard défilable	394
4. Grilles	398
4.1 Paramètres de LazyVerticalGrid et LazyHorizontalGrid	398
4.2 Cas d'usage de LazyVerticalGrid et LazyHorizontalGrid	400
4.2.1 Grille verticale adaptative	400
4.2.2 Grille horizontale avec un nombre de lignes fixe	401
4.3 Grilles contenant des éléments de hauteur ou largeur différente	403
4.3.1 Cas d'usage	403
5. Contrôler et réagir à l'état de défilement	404
5.1 Le State Holder d'un lazy layout	404
5.2 Cas d'usage	406
5.2.1 Observer la position de défilement	406
5.2.2 Actionner le défilement vers une position	411

Chapitre 9

Une interface riche et animée

1. Les composants Materials	413
1.1 Une librairie de composants personnalisables	413
1.2 Le slot API Pattern	415
1.2.1 Définition	415
1.2.2 Principes de conception du Slot API Pattern	415
1.3 Prise en main du composant Scaffold	417
1.3.1 Signature et paramètres	417
1.3.2 Squelette	420
1.3.3 Analyse	421
1.3.4 Cas pratique	422
2. Les animations	430
2.1 Animer la visibilité	430
2.2 Animer la taille d'un contenu	434
2.3 Animer une valeur	437
2.4 Les effets négatifs des animations sur l'accessibilité.	440
2.4.1 Mouvement, clignotement, défilement	440
2.4.2 Animation de mouvement lors d'une interaction	441
2.5 Personnaliser le comportement d'une animation	442
2.5.1 Différents comportements.	442
2.5.2 Accélérer ou ralentir	444
2.5.3 Personnaliser le comportement d'une animation et sa vitesse d'exécution	446
2.6 Aller plus loin	450

12 _____ Jetpack Compose

Développez des interfaces modernes et accessibles pour Android

Chapitre 10

Les tests automatisés

1. Introduction	451
1.1 Installation	451
1.2 Différents types de tests	452
2. Définir l'interface à tester avec createComposeRule	453
3. Les fonctions permettant de tester une interface.	456
3.1 Rechercher, actionner, vérifier.	456
3.2 Matcher.	458
3.3 Associer un tag à un composant	459
4. Tester au sein d'une activité	461
4.1 Accéder aux ressources.	462
4.2 Tester de bout en bout une activité	464
5. Arbre sémantique	465
5.1 Deux versions de l'arbre sémantique : fusionné et non fusionné	465
5.2 Imprimer l'arbre sémantique dans les logs pour déboguer. ...	466
5.3 Cas d'usage : arbre non fusionné dans un test	469
5.4 Synchronisation des tests	473
5.4.1 Cas d'usage : synchroniser manuellement l'interface et contrôler le temps.	474
5.4.2 Cas d'usage : attendre un traitement	477
6. Conclusion	478
 Index	 479